IN THE UNITED STATES PATENT AND TRADEMARK OFFICE

In re Patent Application of

HARRINGTON et al. Atty. Ref.: RWF-3594-14

Serial No. 10/613,319 TC/A.U.: 3694

Filed: 07 July 2003 Examiner: Swartz, Jamie H.

For: PROCESS AND APPARATUS FOR CONDUCTING AUCTIONS OVER ELECTRONIC NETWORKS

* * * * * * * * * * *

Commissioner for Patents
P.O. Box 1450
Alexandria, VA 22313-1450

Sir:

RULE 131 DECLARATION

We, Myles Harrington, Daniel Veres and Robert Panoff, state the following based on our own personal knowledge and belief:

1. We are named as inventors on the patent application identified above.

2. We understand that in a 10/31/2007 Office Action, the Patent Office rejected certain claims as allegedly being "unpatentable" in view of U.S. Patent 5,905,975. We further understand that US Patent No. 5,905,975 was filed on January 2, 1997.

3. We conceived of the subject matter disclosed and claimed in our patent application before January 2, 1997.

4. We attach, as the Exhibit to this declaration, certain documentation that corroborates our conception of the claimed subject matter before January 2, 1997.

5. The attached documentation was created before January 2, 1997, although we have intentionally obscured the date for purposes of filing in the public record.

6. We were diligent in working toward a commercial implementation of the claimed subject matter from before January 2, 1997 through May 29, 1997 when we filed our provisional patent application.

7. We declare further that all statements made herein of our own knowledge are true and that all statements made on information and belief are believed to be true; and further that these statements were made with the knowledge that willful false statements and the like so made are punishable by fine or imprisonment, or both, under Section 1001 of Title 18 of the United States Code and that such willful false statements may jeopardize the validity of the application or any patent issuing thereon.

1/30/08
Date:

Myles Harrington

Date:

Daniel Veres

Date:

Robert Panoff

6. We were diligent in working toward a commercial implementation of the claimed subject matter from before January 2, 1997 through May 29, 1997 when we filed our provisional patent application.

7. We declare further that all statements made herein of our own knowledge are true and that all statements made on information and belief are believed to be true; and further that these statements were made with the knowledge that willful false statements and the like so made are punishable by fine or imprisonment, or both, under Section 1001 of Title 18 of the United States Code and that such willful false statements may jeopardize the validity of the application or any patent issuing thereon.

Date:

Myles Harrington

1/30/08
Date:

Daniel Veres

Date:

Robert Panoff

6. We were diligent in working toward a commercial implementation of the claimed subject matter from before January 2, 1997 through May 29, 1997 when we filed our provisional patent application.

7. We declare further that all statements made herein of our own knowledge are true and that all statements made on information and belief are believed to be true; and further that these statements were made with the knowledge that willful false statements and the like so made are punishable by fine or imprisonment, or both, under Section 1001 of Title 18 of the United States Code and that such willful false statements may jeopardize the validity of the application or any patent issuing thereon.

Date: _____ Myles HARRINGTON

Date: _____ Daniel Veres

Date: 30 Jan '08 Robert Panoff

EXHIBIT

Date: [redacted] 18:24:10 -0800
From: rpanoff@storm.shodor.org (Robert M. Panoff)
To: bob3@shodor.org
Subject: More changes....makes it easier

Bobobob,

Spoke with Myles today for quit a while. He straightened me out on some details and also made some changes.

a) the price which is bid is in fact bid as a percent, not an amount. I already made the various changes to bond.pl to reflect this. No guarantee I made the correct changes, of course!

b) ultimately he would like an "auctioneer" or host page that would look at the bondrate.txt file and compute a total true interest cost for the whole auction, not just year by year. That's pretty simple.
I think...

c) we should add some flag or icon or other indication that tells someone they are the leading bidder at the current time for whatever bids they are submitting. I suggest a 7th column in the table that either has or doesn't have some gif image, or has a separate image for winning vs. lagging.

d) We should enable people to read in their current bids, if they are returning to the auction after leaving.

e) we definitely need to put in a while test that sees if the file is locked.

f) we also need to look at the error logs and understand what is going on:

UX:rm: ERROR: /var/www/htdocs/grantstreet/bondrates.txt not removed: Permission denied.

Are you zapping some fite or something?

He was not able to look at it himself since he does not have the access to the internet from where he called. Maybe tomorrow, he said, when he is back in the office. He did sound enthused over the speed with which this has come together and the description of what is now functional.

Good work! Let me know which, if any, of these things you might have some time for.

Bob

```
#!/usr/bin/perl
# bond.pl
# Bob O'Neill [redacted]

$server_name="storm.shodor.org";
&get_request;
&html_header ("PROPOSAL TO PURCHASE");

# get variables from input
$corporation=$rqpairs{'corporation'};
$id=$rqpairs{'id'};

$price[0]=$rqpairs{'price0'};
$interest[0]=$rqpairs{'interest0'};
$price[1]=$rqpairs{'price1'};
$interest[1]=$rqpairs{'interest1'};
$price[2]=$rqpairs{'price2'};
$interest[2]=$rqpairs{'interest2'};
$price[3]=$rqpairs{'price3'};
$interest[3]=$rqpairs{'interest3'};
$price[4]=$rqpairs{'price4'};
$interest[4]=$rqpairs{'interest4'};
$price[5]=$rqpairs{'price5'};
$interest[5]=$rqpairs{'interest5'};
$price[6]=$rqpairs{'price6'};
$interest[6]=$rqpairs{'interest6'};
$price[7]=$rqpairs{'price7'};
$interest[7]=$rqpairs{'interest7'};
$price[8]=$rqpairs{'price8'};
$interest[8]=$rqpairs{'interest8'};
$price[9]=$rqpairs{'price9'};
$interest[9]=$rqpairs{'interest9'};
$calculate=$rqpairs{'calculate'};
$begin=$rqpairs{'begin'};

# check for a valid ID
open(ID, "/var/www/htdocs/grantstreet/bondid.txt") || die "Couldn't
open bondid.txt\n";
$i=0;
$count=3;
$valid="no";
while ($i < $count) {
  $who=<ID>;
  if ($who =~ $id) {
    $corporation=substr($who, 8, 100);
    chop($corporation);
    $valid="yes";
  }
  $i++;
}
close(ID);

if ($valid eq "no") {
 print "Sorry, Invalid ID\n";
 &html_trailer;
 exit;
 }
```

```
for ($i=0; $i < 10; $i++) {
  if (($interest[$i] eq "" && $price[$i]) || ($price[$i] eq "" &&
$interest[$i])) {
    print "Sorry. blank stuff.\n";
    &html_trailer;
    exit;
    }
  }

# obtain principals from a datafile
open(PRINCIPAL, "/var/www/htdocs/grantstreet/bondprincipal.txt") || die
"Couldn't open bondprincipal.txt\n";

# read in number of years bonds will be offered

     $numyears=<PRINCIPAL>;

for ($i=0; $i < $numyears; $i++) {
     $date[$i]=<PRINCIPAL>;
     $principal[$i]=<PRINCIPAL>;
}

# read in offsets from bond offering and bond sale

      $offset=<PRINCIPAL>;
      $dayfact=<PRINCIPAL>;

close(PRINCIPAL);

for ($i=0; $i < 10; $i++) {
if ($price[$i] eq "") {
  $price[$i]=999999999999;
  $interest[$i]=999999999999;
  }
}


if ($price[0] < 97.5 || $price[1] < 97.5 || $price[2] < 97.5 ||
$price[3] < 97.5 || $price[4] < 97.5 || $price[5] < 97.5 || $price[6] <
97.5 || $price[7] < 97.5 || $price[8] < 97.5 || $price[9] < 97.5) {
  print "Sorry.  At least one of your bids was less than 97.5% of the
corresponding principal.\n";
  print "<p>Use your BACK button to return to the bid submission
page.\n";
  &html_trailer;
  exit;
 }

# compute total of bond principals
$total=0;
for ($i=0; $i < 10; $i++) {
$total=$total+$principal[$i];
}
$total=&commas($total);

# unless we're just beginning,
```

```
unless ($begin ne "") {

# compute yield

#   63 days from end of year until first payment

      $offset=63;

#   76 days from bond offering until first payment, accrued interest

      $dayfact=76;

for ($i=0; $i < 10; $i++) {

if ($interest[$i] < 999) {

  $discount=$price[$i]/100;
  $interest[$i]=$interest[$i]/100;
  $year=$i+1;
  $nsteps=2*($year+1);

# compute first line in table,

      $pvf=.05;
      $sads[1]=0.;
      $timefactor[1]=$offset;
      $proceeds[1]=-$discount;
      $pvfactor[1]=1.;

# precompute second line, since this has accrued interest calculation


      $sads[2]=($interest[$i]/2)*$dayfact/180.;
      $timefactor[2]=$offset/180.;
      $pvfactor[2]=(1+$pvf/2)**(-$timefactor[2]);
      $proceeds[2]=$sads[2];
      $pvproceeds[2]=$pvfactor[2]*$proceeds[2];

# ok, so now do initialization of functions

      $sumproc=0.;

for ($istep=3; $istep < $nsteps+1; $istep++) {
      $sads[$istep]=$interest[$i]/2;
      $timefactor[$istep]=$timefactor[$istep-1]+1.;
      $proceeds[$istep]=$sads[$istep];
      $pvfactor[$istep]=(1+$pvf/2)**(-$timefactor[$istep]);
      $pvproceeds[$istep]=$pvfactor[$istep]*$proceeds[$istep];
      $sumproc=$sumproc+$pvproceeds[$istep];
}

# don't forget you have to pay back the principal sometime

      $proceeds[$nsteps]=$proceeds[$nsteps]+1;
      $sumproc=$sumproc+$pvfactor[$nsteps];

# iterate pvf, jump out when less than a penny error
```

```
      while ((1$proceeds[1]+$sumproc) < -.000001){
#DEBUG
#print "proceeds[1]: $proceeds[1]\n";
#print "sumproc: $sumproc\n";
      $pvf=$pvf*$sumproc/(-$proceeds[1]);
      $sumproc=0.;
for ($j=2; $j < $nsteps+1; $j++) {
      $pvfactor[$j]=(1+$pvf/2)**(-$timefactor[$j]);
      $pvproceeds[$j]=$pvfactor[$j]*$proceeds[$j];
      $sumproc=$sumproc+$pvproceeds[$j];
      }
}

      $yield[$i]=100*$pvf;
  }
}


for ($i=0; $i < 10; $i++) {
if ($interest[$i] < 999) {
  $interest[$i]=$interest[$i]*100;
  }
}

# read in the best yield
open (BEST, "/var/www/htdocs/grantstreet/bestrates.txt") || die
"Couldn't
open bestrates.txt\n";
 for ($i=0; $i < 10; $i++) {
  chop($idwin[$i]=<BEST>);
  chop($pricewin[$i]=<BEST>);
  chop($interestwin[$i]=<BEST>);
  chop($bestyield[$i]=<BEST>);
}
close(BEST);

# skip comparison if calculate is set

if (!$calculate){

# see if the yield rates are the lowest

for ($i=0; $i < 10; $i++) {

if ($interest[$i] < 999) {

if ($yield[$i] < $bestyield[$i]) {
  $bestyield[$i]=$yield[$i];
  $idwin[$i]=$id;
  $pricewin[$i]=$price[$i];
  $interestwin[$i]=$interest[$i];
   }
  }
 }
}
#DEBUG
```

```
#don't think we need this anymore
#for ($i=0; $i < 10; $i++) {
#   $yield[$i] = substr($yield[$i], 0, 5);
#   $bestyield[$i] = substr($bestyield[$i], 0, 5);
#}

# see if bestrates.txt is in use
#$temp="LOCKED";
#while ($temp eq "LOCKED") {
#open (RATELOCK, "/var/www/htdocs/grantstreet/bestrateslock.txt") ||
die
"Couldn't open bestrateslock.txt\n"; #chop($temp=<RATELOCK>);
#close(RATELOCK);
#}

# write out the new best rates
open (BEST, ">/var/www/htdocs/grantstreet/bestrates.txt") || die
"Couldn't
open bestrates.txt the second time\n"; for ($i=0; $i<10; $i++) {
  printf BEST "%.3f\n", $idwin[$i];
  printf BEST "%.3f\n", $pricewin[$i];
  printf BEST "%.3f\n", $interestwin[$i];
  printf BEST "%.3f\n", $bestyield[$i];
}
close(BEST);

# unlock bestrates.txt
#open (RATELOCK, ">/var/www/htdocs/grantstreet/bestrateslock.txt") ||
die
"Couldn't open bestrateslock.txt the second time.\n";
#print RATELOCK "UNLOCKED\n";
#close(RATELOCK);

# put commas in bond principals
for ($i=0; $i<10; $i++) {
$principal[$i]=&commas($principal[$i]);
}

# see if bondbids.txt is in use
#$tmp="LOCKED";
#while($tmp eq "LOCKED") {
#open (BIDLOCK, "/var/www/htdocs/grantstreet/bondbidlock.txt") || die
"Couldn't open bondbidlock.txt\n";
#chop($temp=<BIDLOCK>);
#close(BIDLOCK);
#}

# if not, lock it
#open (BIDLOCK, ">/var/www/htdocs/grantstreet/bondbidlock.txt") || die
"Couldn't open bondbidlock.txt the second time.\n";
#print BIDLOCK "LOCKED\n";
#close(BIDLOCK);

# write out the bids to a file
open (BIDS, ">>/var/www/htdocs/grantstreet/bondbids.txt") || die
"Couldn't open bondbids.txt\n";
```

```
for ($i=0; $i<10; $i++) {
$yield[$i] = substr($yield[$i], 0, 5);
}


for ($i=0; $i<10; $i++) {
  if ($interest[$i] > 99998) {
     $price[$i]="NO BID";
     $interest[$i]="NO BID";
     $yield[$i]="NO BID";
     }
}

# for ($i=0; $i<10; $i++) {
#$bestyield[$i] = substr($bestyield[$i], 0, 5);
#}

unless ($interest[0] eq "NO BID" && $interest[1] eq "NO BID" &&
$interest[2] eq "NO BID" && $interest[3] eq "NO BID" && $interest[4] eq
"NO BID" && $interest[5] eq "NO BID" && $interest[6] eq "NO BID" &&
$interest[7] eq "NO BID" && $interest[8] eq "NO BID" && $interest[9] eq
"NO BID") {
print BIDS "$id ($corporation):\n";

for ($i=0; $i<10; $i++) {
  if ($price[$i] =~ /NO/) {
     print BIDS "$price[$i]\t $interest[$i]\t $yield[$i]\n";
     }
  else {
     printf BIDS "%.3f\t", $price[$i];
     printf BIDS "%.3f\t", $interest[$i];
     printf BIDS "%.3f\n", $yield[$i];
     }
}
print BIDS "\n";
}
close(BIDS);

# now change NO BIDs back to blank
for ($i=0; $i<10; $i++) {
if ($interest[$i] =~ /NO BID/) {
  $price[$i]="";
  $interest[$i]="";
  $yield[$i]="";
  }
}

# write out last-submitted values
  if (!$calculate) {
     open (LAST, ">/var/www/htdocs/grantstreet/$id.txt") || die
"Couldn't open
     $id.txt\n";
     for ($i=0; $i < 10; $i++) {
       printf LAST "%.3f\n", $price[$i];
       printf LAST "%.3f\n", $interest[$i];
       printf LAST "%.3f\n", $yield[$i];
     }
```

```
  }

} # end of "unless we're just beginning"
else { # if we ARE just beginning
  # read in the best yield
  open (BEST, "/var/www/htdocs/grantstreet/bestrates.txt") || die
"Couldn't
  open bestrates.txt\n";
  for ($i=0; $i < 10; $i++) {
    chop($idwin[$i]=<BEST>);
    chop($pricewin[$i]=<BEST>);
    chop($interestwin[$i]=<BEST>);
    chop($bestyield[$i]=<BEST>);
  }
  close(BEST);

  # read in last-submitted values for $id
  open (LAST, "/var/www/htdocs/grantstreet/$id.txt") || die "Couldn't
open
  $id.txt\n";
  for ($i=0; $i < 10; $i++) {
    chop($price[$i]=<LAST>);
    chop($interest[$i]=<LAST>);
    chop($yield[$i]=<LAST>);
  }
for ($i=0; $i < 10; $i++) {
  $principal[$i]=&commas($principal[$i]);
 }
}

# print out the new form
print <<EOI;
<center>\$$total<sup>*</sup><br>
City of Pittsburgh, Pennsylvania<br>
General Obligation Refunding Bonds, Series A of 1996<br></center>

<form method="POST"
action="http://storm.shodor.org/cgi-bin/grantstreet/bond.pl">

<input type="hidden" name="corporation" value="$corporation">
<input type="hidden" name="id" value="$id">

<table border=0 cellspacing=5 cellpadding=5>
<tr><th>Due<br>March 1<th>Principal<br>Amount<sup>*</sup><th>Price
<th>Interest<br>Rate<th>Yield<th>Best Yield
EOI


for ($i=0; $i<10; $i++) {

print <<EOI;
<tr>
<td align=center>$date[$i]</td>
<td align=center>\$$principal[$i]</td>
<td align=center><input name="price$i" value="
EOI
```

```
printf ("%.3f",$price[$i]) if ($price[$i] !="");

print <<EOI;
" size=7 maxlength=11>\%</td>
<td align=center><input name="interest$i" value="
EOI

printf ("%.3f",$interest[$i]) if ($interest[$i] !="");

print <<EOI;
" size=5 maxlength=5>\%</td>
<td align=center>
EOI

printf ("%.3f%",$yield[$i]) if ($yield[$i] !="");

print <<EOI;
<td align=center>
EOI

printf ("%.3f%",$bestyield[$i]);

print <<EOI;
</td>
<td align=center>
EOI
  if ($idwin[$i] eq $id) {
    print "you win\n";
  }
  else {
    print "you lose\n";
  }
print <<EOI;
</td>
</tr>
EOI
} # end of for (0<=i<10)

print <<EOI;

</table>
Name of firm submitting Proposal: <b>$corporation</b><br>
<input type=submit name="calculate" value="Calculate / Refresh">
<input type=submit value="Submit Bids"><br>
<sup>*</sup>Preliminary, subject to change
<p><hr>
EOI

# unlock bondbids.txt
#system("rm /var/www/htdocs/grantstreet/bondbidlock.txt");
#open (BIDLOCK, ">/var/www/htdocs/grantstreet/bondbidlock.txt") ||
"Couldn't open bondbidlock.txt the third time\n";
#print BIDLOCK "UNLOCKED\n";
#close (BIDLOCK);
#system("chmod a+w /var/www/htdocs/grantstreet/bondbidlock.txt");

&html_trailer;
```

```
exit;

sub round {
    local($number) = @_;
    $roundednumber=1000*$number;
     $fraction=$roundednumber-int($roundednumber);
    if ($fraction >= 0.5) {
      $roundednumber++;
    }
    $roundednumber/=1000;
    $roundednumber=substr($roundednumber,0,5);
    return $roundednumber;
}

sub commas {
    local ($_)=@_;
    1 while
    s/(.*\d)(\d\d\d)/$1,$2/;
    $_;
}

sub get_request {


     if ($ENV{'REQUEST_METHOD'} eq "POST") {
        read(STDIN, $request, $ENV{'CONTENT_LENGTH'});
     }
        else
        {
                print "Location: http://$server_name/grantstreet/\n\n";
                exit;
        }

    %rqpairs = &url_decode(split(/[&=]/, $request));
}

sub url_decode {

#       Decode a URL encoded string or array of strings
#               + -> space
#               %xx -> character xx

    foreach (@_) {
        tr/+/ /;
        s/%(..)/pack("c",hex($1))/ge;
    }
    @_;
}

sub html_header {

    # Subroutine html_header sends to Standard Output the necessary
    # material to form an HTML header for the document to be
    # returned, the single argument is the TITLE field.

    local($title) = @_;
```

```
    print "HTTP/1.0 200 OK\n";
    print "MIME-Version: 1.0\n";
    print "Server: WebSTAR/1.2.5\n";
    print "Content-type: text/html\n\n";
    print "<html><head>\n";
    print "<title>$title</title>\n";
    print "</head><body bgcolor=\"#ffffff\">\n";
    print "<center><h1>$title</h1></center>\n";
}

sub html_trailer {
        print "<p><a href=\"http://storm.shodor.org/grantstreet\"><img
border=0 align=center";
        print " alt=\"\" src=\"../../grantstreet/gsabutton.gif\">";
        print " Grant Street Advisors</a>\n";
        print "<hr>\n";
        print "<address>Please direct questions and comments to ";
        print "<a href=\"mailto:webmaster\@shodor.org\">";
        print "webmaster\@shodor.org</a><br>\n";
        print "Copyright &copy [redacted] Grant Street Advisors";
        print "</address>\n";
        print "</body></html>\n";
}
```